DWIT COLLEGE

DEERWALK INSTITUTE OF TECHNOLOGY

CONTENT BASED AUTOMATIC MUSIC GENRE CLASSIFICATION USING FFT AND MFCC

A MINI PROJECT REPORT

Submitted to

Department of Computer Science

DWIT College

Submitted by

Ruby Shrestha

December, 2016

DWIT College

DEERWALK INSTITUTE OF TECHNOLOGY

SUPERVISOR'S RECOMMENDATION

I hereby recommend that this project prepared under my supervision by RUBY SHRESTHA entitled **"CONTENT BASED AUTOMATIC MUSIC GENRE CLASSIFICATION USING FFT AND MFCC"** in partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Information Technology be processed for the evaluation.

……………………………………………

Basanta Joshi

Lecturer

Deerwalk Institute of Technology

DWIT College

DWIT College

DEERWALK INSTITUTE OF TECHNOLOGY

STUDENT'S DECLARATION

I hereby declare that I am the only author of this work and that no sources other than that listed here have been used in this work.

……………………………………

Ruby Shrestha

Date:- December, 2016

DWIT College

DEERWALK INSTITUTE OF TECHNOLOGY

LETTER OF APPROVAL

This is to certify that this project prepared by RUBY SHRESTHA entitled **"CONTENT BASED AUTOMATIC MUSIC GENRE CLASSIFICATION USING FFT AND MFCC"** in partial fulfillment of the requirements for the sixth semester of degree of B.Sc. in Computer Science and Information Technology has been well studied. In our opinion it is satisfactory in the scope and quality as a project for the required degree.

…………………………………… [Supervisor] Lecturer DWIT College	……………………………………… Chief Academic Officer DWIT College
………………………………………….. [External Examiner] Academic Designation IOST, Tribhuvan University	………………………………………….. [Internal Examiner] Assistant Professor IOST, Tribhuvan University

ACKNOWLEDGEMENT

I would like to thank Dr. Basanta Joshi, Assistant Professor, IOE, TU for providing technical help and providing deeper knowledge about the technology needed for this project. Without his help, I would be struggling to understand the theory needed and might have reached an incorrect conclusion.

I would like to thank Ms. Shweta Shrestha and Mr. Pramesh Gautam for assisting me during this project. They have mentored me and helped me follow the right path for the project.

At last but not the least, my sincere thanks goes to my parents and members of my family, who have always supported me, and to all of my friends who directly or indirectly helped me to complete this project report.

Ruby Shrestha
Exam Roll no: 0341

Batch 2017A

ABSTRACT

Music genres are categories that can be used to identify and arrange growing amount of music emerging at present. Conventionally, music genre labeling was done manually. However, due to the advent of new techniques and growing number of researches in Music Information Retrieval, some form of automation has been seen in the field of music genre categorization. In this project, a stepping stone for automatic music genre categorization of vast number of music files available in digital form online or offline has been developed. Out of the various techniques of music genre recognition, content based technique has been used to automatically label the particular user uploaded song into one of five distinct genres: Classical, Pop, Metal, Jazz, and Blues. Digital signal processing techniques of Fast Fourier Transform and Mel-Frequency Cepstral Coefficients have been used to generate feature values of feature vector, which is then fed into the classifier developed using Support Vector Machine, in order to classify user input song. The training and testing of the system has been performed successfully obtaining an accuracy of 74.0%, which is significant in the field of music analysis. Training has been carried out using 80 music clips of each of the five genres, and testing, using 20 music clips of each of the five genres. GTZAN Music Dataset, a popular Western music dataset prepared for music analysis, has been used during training and testing; hence, this system works well only with Western music files. The system has been implemented with two graphical user interfaces: one for admin (training and testing) and another for user (uploading music file and finding genre) using Python programming language and Flask framework.

Keywords: Automatic Music Genre Classification, Fast Fourier Transform, Mel-Frequency Cepstral Coefficients, Support Vector Machine, Music Information Retrieval

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AMGC: Automatic Music Genre Classification

SVM: Support Vector Machine

GMM: Gaussian Mixture Model

LDA: Linear Discriminant Analysis

FFT: Fast Fourier Transform

MFCC: Mel-Frequency Cepstral Coefficients

MIR: Music Information Retrieval

DCT: Discrete Cosine Transform

DWT: Discrete Wavelet Transform

ASA: Acoustical Society of America

CSS: Cascading Style Sheet

HTML: Hyper Text Markup Language

URL: Uniform Resource Locator

Note: The system has been referred to as "Automatic Music Genre Classifier" in the rest of the document.

CHAPTER 1: INTRODUCTION

1.1 Background

Music is an organized sound. It is distinguished from other organized sounds such as human speech and sounds produced by animals based on musical features such as pitch (that indicates melody and harmony), timbre (quality of musical note distinguishing different sound productions such as voices and musical, string instruments), rhythm (related to sound meter and tempo), and the dynamics of loudness and softness. (Stanford, 2012) Not all of these elements are available in all kinds of music; some lay emphasis on one combination of elements while others on a different combination. This difference leads to the concept of different music styles. The presence of different music styles in turn leads to the notion of different musical genres.

Musical genres are connected to emotional, cultural and social aspects, and all of them influence our music understanding. As mentioned by Franco Fabbri in his paper "A THEORY OF MUSICAL GENRES: TWO APPLICATIONS", a musical genre is "a set of musical events (real or possible) whose course is governed by a definite set of socially accepted rules". These socially accepted rules can contain technical as well as non-technical aspects. (Fabbri, 1980)

Genres can be characterized using the following types of rules, of which only the first is related strictly to musical content:

• Formal and technical: Content-based practices.

• Semiotic: Abstract concepts that are communicated (e.g., emotions or political messages).

• Behavior: How composers, performers and audiences appear and behave.

• Social and ideological: The links between genres and demographics such as age, race, sex and political viewpoints.

• Economical and juridical: The laws and economic systems supporting a genre, such as record contracts or performance locales (e.g., cafés or auditoriums).

These different types of rules contribute to the definition of a genre and on the ways in which they are accepted by various communities.

(McKay & Fujinaga, 2006)

Until now, music has been assigned genre labels manually, since automation in the field of musical genres is still a field of research. Automation in music genre classification relies on Music Information Retrieval (MIR). A commonly-attempted task in MIR is to automatically associate a genre label with a given track, with primary foundation on its audio content. MIR has its basis on signal analysis. One way to deal with this is to regard it as a classification problem, and to model genres as clusters in some feature space; by taking in the parameters of those clusters from labeled training data, one can then allot genre to an obscure track. (Music Information Retrieval)

1.2 Overview

With the advent of Internet and technologies based on it, people can now download and upload large collection of music files. Consequently, the online music database is vast and will be ever growing. Efficient and accurate automatic music information processing (accessing and retrieval, in particular) will be an extremely important issue. (Li, Ogihara, & Li, 2003) Such music information processing can lead to significant outcomes such as knowledge of music styles of corresponding music. MIR plays a significant role in such knowledge retrieval and one of the preferred goals of MIR is to organize and categorize music contents in web.

A 2004 survey by Lee JH and Downie JS concluded that end users are more likely to browse and search by genre rather than by recommendation, artist similarity or music similarity, although these alternatives were each popular as well. (Lee & Downie, 2004) Hence, as implied by this survey result, categorization by genre seems to overshadow other categorizations. However, when the number of files increases, such categorization becomes a tedious job if done manually, and many a times, we fail to recognize a song's genre.

Currently, although tedious, music genre classification is done mainly by hand because giving a precise definition of a music genre is extremely difficult and, in addition, many music sounds sit on boundaries between genres. These difficulties are due to the fact that

music is an art that evolves, where performers and composers have been influenced by music in other genres. (Li, Ogihara, & Li, 2003)

Genre plays an important role in music domain. A study by North AC and Hargreaves DJ. indicates that genre is so important to listeners that the style of a piece can influence their liking for it more than the piece itself.

Therefore, music genre classification is a growing field that has prospect as well as opportunity- prospect because genre has a vital purpose in music domain and opportunity resulted by the problem that manual music genre classification is a time consuming and daunting task.

Automating Music Genre Classification (AMGC) will be bliss, and this bliss is not impossible. It has been observed that audio signals (digital or analog) of music belonging to the same genre share certain characteristics, because they are composed of similar types of instruments, having similar rhythmic patterns, and similar pitch distributions. This suggests feasibility of automatic musical genre classification. (Li, Ogihara, & Li, 2003)

AMGC has a wide prospect in real world as well. Consider the following:

- Shop owners/ restaurants seek music of particular genre to attract customers.
- Advertisements look for songs of particular genre to make use of.
- A Disk Jockey (DJ) seeks music with similar tempo/beat (genre).
- Genre thematic dance parties are increasing.

These facts implicitly or explicitly indicate that AMGC can indeed be significant in 21st century.

Automatically classifying the vast online music database that we can create with music content on web requires considerable technical and schedule based resources. This project, hence, desires to build a stepping stone for content-based AMGC of such vast music database by enabling the users to automatically classify the song uploaded by him/her into one of five genres:

- Classical
- Pop
- Metal
- Blues

3

- Jazz

The five genres have been chosen solely by self interest. Also, these genres are five of the popular music genres of 21st century and are distinguishable from each other.

AMGC involves process of music feature extraction followed by classification procedure. Music feature extraction involves taking analog signal, changing it into raw data, extracting and selecting feature and producing feature vector. Popular music features that have been used for AMGC in prior researches include FFT, MFCC, Beat, Pitch, Tempo, Loudness and similar. The feature vector is then used in classification algorithm for producing classification. Popular classification algorithms include SVM, K-Nearest Neighbors, and Neural Networks.

This project uses FFT and MFCC for feature extraction and SVM as classification algorithm.

1.3 Problem Statement

Music has been evolving since prehistoric age. The progression of music has given rise to a number of genres. Genres are considered to be an important element of music domain and is said to influence the liking of people towards music. Internet holds a large collection of music of different genres due to emergence and advancement of different technologies, ease of uploading and downloading music. Such large collection requires organization and categorization. However, when such process of categorization is done manually, it is time consuming.

Automating such categorization can ease the process and lead to significant minimization in the effort required by people in the process. Also, AMGC is an advancing field of research gaining popularity. Considering these, this project focuses on implementing the process of content-based AMGC and classifying user input song into one of five distinguishable genres (classical, pop, metal, jazz, blues).

1.4 Objectives

1.4.1 General objective:

To implement the process of content-based AMGC using Fast Fourier Transform and Mel-Frequency Cepstral Coefficient for feature extraction and SVM for classification.

1.4.2 Specific objective:

a) To manage data and preprocess it into format required

b) To match the feature of user input song with stored features to determine the input song's genre label and determine accuracy

1.5 Scope

The project will classify music based on genre. Specifically, the system developed will classify only Western songs suitably because of the easy availability of the popular music dataset (GTZAN Music Dataset) for Western songs, considered as training data in the system. Moreover, classification works well only on five distinguishable genres as of now, namely, Classical, Pop, Metal, Blues, and Jazz. The music file formats that the user can upload to find its genre label has been limited to mp3, wav, aiff, m4a, and au for ease.

1.6 Limitation

1. Music uploaded by user will be labeled as one of five genres Classical, Pop, Metal, Blues, and Jazz based on feature similarity although its actual genre may not be any of the five considered genres (Classical, Pop, Metal, Blues, and Jazz).

2. When the user uploads music file of format besides mp3, wav, aiff, m4a, and au for classification, analysis will not be carried out; s/he will be prompted that the system does not handle the respective file format.

3. Since the system has been trained and tested using monographic music clips, the analysis and result of stereographic music files might not be as expected or will have significant chances of being incorrect.

4. Since the system has been trained and tested using Western music clips, the analysis and result of non-Western music files might not be as expected or will have significant chances of being incorrect.

1.7 Outline of Document

The report is organized as follows

| Preliminary Section	• Title Page • Abstract • Table of Contents • List of figures and Tables
Introduction Section	• Background • Problem Statement • Objectives • Scope • Limitation
Requirement and Feasibility Analysis Section	• Literature Review • Requirement Analysis • Feasibility Analysis
System Design Section	• Methodology • Algorithm • System Design
Implementation and Testing Section	• Implementation • Description of Major Classes • Testing
Maintenance and Support Plan Section	• Maintenance Plan • Support Plan
Conclusion and Recommendation Section	• Conclusion • Recommendation

CHAPTER 2: REQUIREMENT AND FEASIBILITY ANALYSIS

2.1 Literature Review

2.1.1 Human Music Perception

Music comes under sound. In order to understand how human auditory system understands music, first we need to understand how it perceives sound. How human ear perceives sound is a psycho-acoustical phenomenon.

Human brain and human ear work in cohesion to understand musical feature. Vibrating ear molecules coming in contact with human ear drum at different frequencies and velocities leads to the production of sound. Human sound perception starts at the human ear and ends at the human brain.
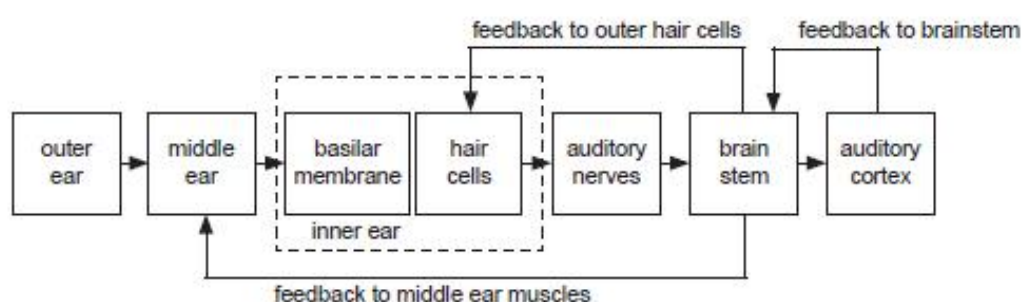
Figure 1 - Schematic diagram of human auditory perception

Human ear consists of three major parts: outer ear, middle ear and inner ear. The process of auditory perception starts with sound waves hitting the outer ear. Outer ear channels these vibrations to the inner ear through the middle ear, which converts acoustic vibration to mechanical vibration. Middle ear connects outer ear to inner ear (oval opening window of inner ear specifically) via its three interconnected ear bones called ossicles. Inner ear consists of important elements called cochlea and basilar membrane. Cochlea consists of fluid which moves back and forth on receiving the vibration. This causes motion in the basilar membrane, which consists of approximately twenty-thousand hair-like nerve cells, each of which has a natural sensitivity to a particular resonant frequency. A nerve impulse is generated when those hair cells are excited by vibration. That nerve impulse is sent to brain. (Kosina, 2002)

Brain's primary auditory cortex receives signal from inner ear through brain stem. The primitive brain, cerebellum, is then activated. Ear and cerebellum are collectively called low level processing unit. They perform the principle feature extraction which permits the brain to begin dissecting the sounds, separating sensory stimulus into pitch, timbre, spatial area, sufficiency, tone lengths, and onset times of various notes. Neurons carry this data to high level processing units in the frontal brain lobe. Brain hence sorts sound into music through the collaboration between the low-level and high level processing units. (PPL, PRS)

2.1.2 Music Information Retrieval – Application and Strategies

Music Information Retrieval is the retrieval of information from music. It includes strategies for enabling access to music collection, both new and historical. In the area of Music Information Retrieval (MIR), great technical progress has been made since this discipline started to mature in the late 1990s. (Wiering, 2006) MIR today is the basis of various research and technical applications. Some common tasks undertaken in the field of MIR is shown by the figure below. (Music Information Retrieval)
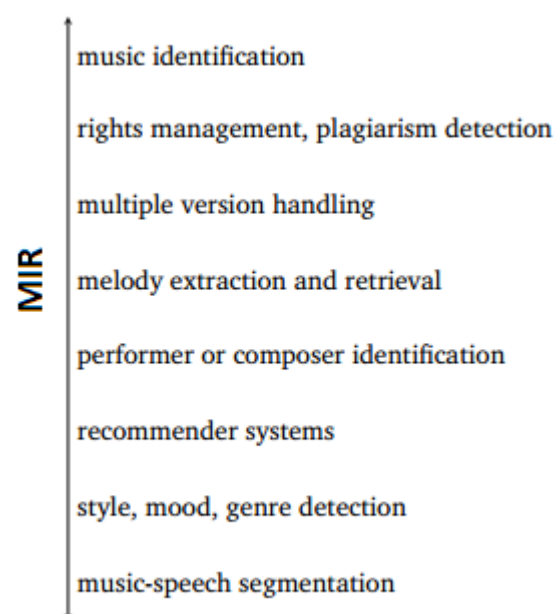
Figure 2 – Enumeration of tasks in the general field of Music Information Retrieval

At present, various strategies are being employed in the domain of MIR, three basic strategies being the following:

1. Textual metadata approach
   Music metadata includes title of the song, band or artist's name, album name that the song originates from, type of music (genre), album track number, year the song was released.

2. Approach based on high level descriptions
   High level descriptions can include timbre, melody, rhythm, pitch, harmony, key, structure.

3. Approach based on low level signal descriptions
   Low level signal based descriptions include information in the digital audio.

(Casey, 2008)

Because there is a great number of music recordings available that can be used as a first stage input to a high level music description system, this motivates work on extracting high-level music features from low level audio content.

At present, most of the MIR applications make use of low level descriptions rather than high level music descriptions, in contrast to human music perception which tend to rely on high level descriptions. (Kaminskas & Ricci, 2012)

Music classification by type/genre is one of such MIR applications that make use low level descriptions as reflected by different research works done till date.

2.1.3 Related Works -Automatic Music Genre Classification

Synopsis

Figure 2 explicitly lists style, mood and genre detection as one of the areas in which MIR is used. Consequently, AMGC is an MIR application.

Musical genres can be quite subjective making automatic classification difficult. Until the beginning of 2000s, although manually annotated genre information had been used to evaluate content-based similarity retrieval algorithms, there had been no prior published work in automatic genre classification. From the beginning of 2000s, researches and

works on AMGC started with one of the first works in the field being "Automatic Musical Genre Classification of Audio Signals" done by George Tzanetakis, Georg Essl and Perry Cook in 2001. (Tzanetakis, Essl, & Cook, 2001) Tzanetakis and Cook published next paper in the same domain in 2002 with the then-significant classification accuracy of 61%. (Tzanetakis & Cook, 2002) Considering these works as foundations, many other works in the field then started gradually increasing.

Features

AMGC includes a phase of feature extraction which produces feature vector. The feature vector is later fed to classification algorithm during classification phase.

What music feature(s) one considers is a vital thing to keep note of when carrying out AMGC since the success of AMGC not only relies on the classification algorithm used, but also on what features one has employed when carrying out classification.

Different researchers have considered different approaches for music feature extraction in the process of AMGC, most of the feature extraction being based on low level descriptions.

The study by Beth Logan in Mel-Frequency Cepstral Coefficients for Music Modeling demonstrates the use of MFCC for music classification. (Logan, 2000)

One of the classic and most cited works in the area of AMGC is the one of Tzanetakis and Cook. In the research work, Tzanetakis and Cook have used Short-time Fourier Transform (STFT), Mel-Frequency Cepstral Coefficients (MFCCs), and some additional parameters to obtain three feature vectors representing timbral texture, rhythmic and pitch content. Those feature vectors were then used to train statistical pattern recognition classifiers such as simple Gaussian, Gaussian Mixture Model, and k-Nearest Neighbor and an accuracy of 61% was obtained on 10 music genres. (Tzanetakis & Cook, 2002)

Debora C Correa, Luciano da F Costa, and Jose H Saito. in "Tracking the Beat", handled music stored in the discrete MIDI format and used weighted digraphs to model the rhythms dominant in different genres. (Correa, Costa, & Saito, 2010)

Furthermore, the researchers who had researched on Audio Feature Engineering for AMGC concluded that their accuracy increased from 80.5% to 82.3% when considering advanced features like reverse, chorus, beat, and pitch along with basic features (FFT,

MFCC, volume, centroid) instead of using only basic features. (Annesi, Basili, Gitto, & Moschitti)

To add, in "A Comparative Study on Content-Based Music Genre Classification", when Music Genre Classification was carried out using pitch values, beat values, FFT, and MFCC separately on particular dataset using different classifiers, accuracies of around 36%, 40%, 57%, and 60% respectively were obtained. When FFT and MFCC were combined for music feature on the same dataset for different classifiers, the accuracy increased to 65%. (Li, Ogihara, & Li, 2003)

In the book "Building Machine Learning Systems with Python", the frames of MFCC so obtained were averaged over 13 coefficients to get a set of 13 averaged coefficients as feature vector. The averaging was done to not overwhelm the classifier during classification with heavy feature vector considering each MFCC value. (Coelho & Richert)

Classifier

Different popular supervised machine learning algorithms have been used in the past in order to automate the process of music genre classification.

Karin Kosina performed AMGC using KNN with the features distinguished into two categories: features related to the musical texture and features related to the rhythm/beat of the sound. A k-nearest-neighbor classifier is trained with a set of sample data consisting of randomly chosen musical excerpts of 3 seconds length. The system reaches a classification accuracy of 88 % for the three test genres Metal, Dance, and Classical. (Kosina, 2002)

Support Vector Machine (SVM) was used on Magnatune 2004 corpus and RTV corpus in the process of audio feature engineering. With SVM, the researchers had predicted that when considering advanced features like reverse, chorus, beat, and pitch along with basic features (FFT, MFCC, volume, centroid), instead of using only basic features, accuracy increased by two absolute percents. (Annesi, Basili, Gitto, & Moschitti)

Antonio Jose Homsi Goulart, Rodrigo Capobianco Guido, and Carlos Dias Maciel in "Exploring different approaches for music genre classification", made use of combined SVMs. (Goulart, Guido, & Maciel, 2012) The first type of classifier was based on the

11

training of three SVMs in which each one was trained to return value 1 in case of blues, classical or lounge, respectively, and -1 for the other cases. The first was trained to return 1 in case of classical music, the second was trained to return 1 in case of classical music, and the third one in case of lounge music. The second type of classifier also used three SVMs, but each was trained to return 1 in case of recognition of its genre, never returning -1. Feature vector consisting of entropy values in the time domain and feature vector consisting of entropy values in the frequency domain were used separately in each kind of test. Better results were obtained with frequency values and with the second kind of SVM architecture.

In a paper titled "A Comparative Study on Content-Based Music Genre Classification", music genre classification was carried out using different training algorithms, namely, SVM (one-versus-one and one-versus-all), GMM, KNN, and LDA. SVMs resulted in better accuracy than other training algorithms. (Li, Ogihara, & Li, 2003)

2.2 Requirement Analysis

The requirement analysis for this project is broken down into functional and non-functional requirements.

2.2.1 High Level Requirement

Since the system to be developed is a music classification system, it requires classifying any given music file into one of the following five genres:

a) Classical
b) Pop
c) Metal
d) Jazz
e) Blues

2.2.2 Functional Requirements

a) Implement supervised machine learning algorithm to train the machine and help it recognize various music files of different genre by feeding feature matrix and corresponding labels into it.

b) Save the trained model so that the system does not require training time and again.

c) Test the trained model

d) Develop a user module where user can get to know genre of the input song.

2.2.3 Non-Functional Requirements

The corresponding non-functional requirements of the functional requirements are as follows:

a) Support Vector Machine (SVM), a popular supervised machine learning algorithm that is based on support vectors, will be used for training. Feature matrix for training will be formed using FFT and MFCC values of the wav files in the training set. 80-20 rule is used for machine learning, meaning 80% of the total data collected will be used for training and 20% will be used for testing.

b) The trained model will be saved in the local file storage so that it can later be accessed and we can apply our prediction method to it.

c) 20% of the total available music dataset will be used for testing.

d) A web interface will be developed to enable the user to input song whose genre s/he would like to know. Web interface will be developed in Flask. The web interface will contain an upload button through which user can upload an image. And another button "Predict" which runs the testing algorithm internally and renders the result in a web page. The rendered result will also have placeholders to show the extracted features of the image and associated graphical plots.

Other non-functional requirements include code quality, performance, fault tolerance, scalability, testability and maintainability.

2.3 Feasibility Analysis

2.3.1 Technical feasibility

Automatic Music Genre Classifier requires a programming language that can deal with music files easily. Programming languages such as Java and Python are available for the purpose. Furthermore, since it is going to be a web application, a web application framework compatible with the programming language is required.

The project is technically feasible as it can be built using the existing available technologies. The tools and modules needed to build the system are open source, freely available and are easy to use.

Python, as programming language, and Flask, as programming framework, have been chosen.

2.3.2 Operational feasibility

With the availability of Internet and browser in the machine, any user can easily operate the application for classifying the input music file into genre. The application does not require any special hardware besides computer system. Hence, the project is operationally feasible.

2.3.3 Schedule feasibility

Critical Path Method (CPM) is used to carry out schedule feasibility. Critical tasks and their interrelationship are identified to plan the project to prevent time-frame problems and process bottlenecks.

Table 1 - Schedule for Automatic Music Genre Classifier

Tasks	Time (days)	Predecessors
Data Collection (A)	2	-
Literature Review - Feature Extraction (B)	4	-
Literature Review - Classification (C)	4	B
Data/File Formatting (D)	1	A

	6	B, D
Feature Extraction (E)		
Training (F)	5	C, E
Testing (G)	5	E, F
User Module (H)	6	F, G
UI Design (I)	4	H
Documentation (J)	12	A,D,E,F,G,H
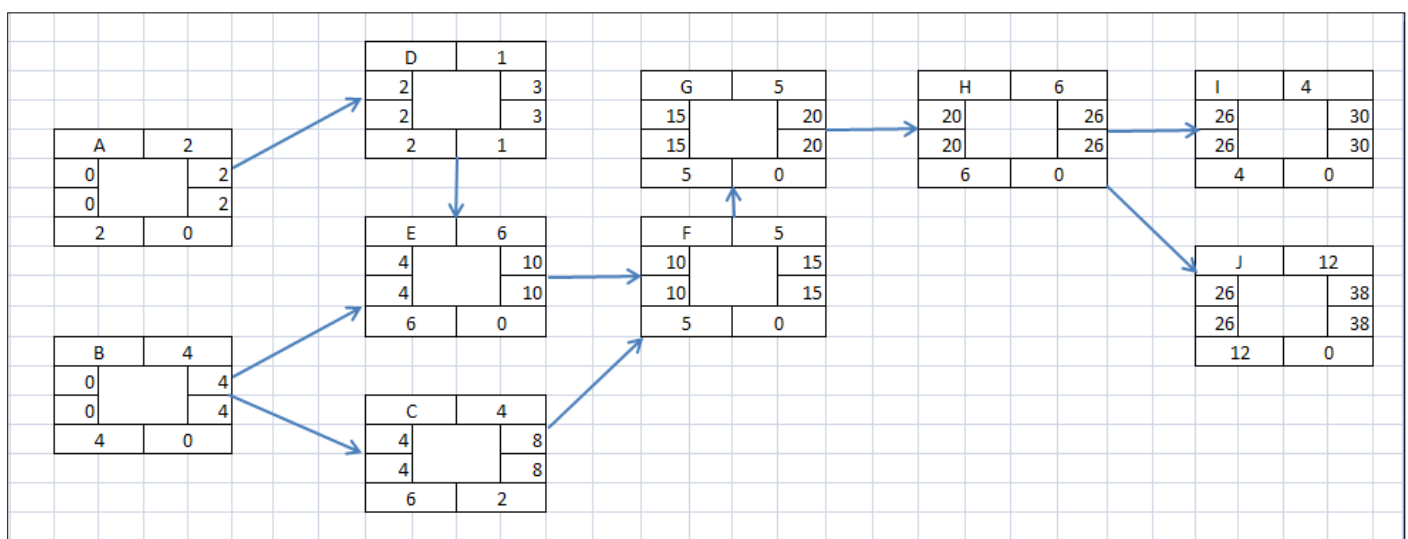
EST- Early Start Time

LST- Late Start Time

EFT- Early Finish Time

LFT- Late Finish Time

Task		Duration
EST		LST
EFT		LFT
Span		Float

Float = EFT-EST

Span = Duration + Float

Figure 3 - Activity Network Diagram of the System

Figure 3 shows the Activity Network Diagram of the system. The total (estimated) days required for the completion of the project is 38 days, which is almost equal to five and half weeks. This is less than half of the total working days of a semester, which is almost four months. Hence, the project was determined to be feasible in terms of schedule.

CHAPTER 3: SYSTEM DESIGN

3.1 Methodology

Different comparative researches performed concerning the methods for AMGC are inclined towards the use of SVM. They show that the accuracy of the system is comparatively better with the use of SVM (as mentioned in Literature Review). Therefore, the proposed system (Automatic Music Genre Classifier) makes use of SVM (one-versus-one/pairwise approach) with the feature vector consisting of FFT and MFCC values of each music file present in the traning music dataset. FFT and MFCC indicate spectral energy and timbral feature of the music files respectively. The system is implemented using Python programming language and Flask web framework.

3.1.1 Data collection

The dataset that Automatic Music Genre Classifier uses is the GTZAN music dataset. This dataset contains 30 seconds clips, in .au format, of 100 songs each in 10 music genres. This dataset was prepared towards the beginning of 21st century and hence, it is not very old. It has been used by many famous researches in the field of music classification, an exemplary one being Automatic Musical Genre Classification of Audio Signals by George Tzanetakis, Georg Essl and Perry Cook.

3.1.2 Data selection

Out of the music files present in 10 music genres (Classical, Pop, Hiphop, Country, Disco, Metal, Jazz, Pop, Reggae, and Blues) in GTZAN music dataset, the project uses the music files present in only five distinct music genres (Classical, Pop, Metal, Jazz and Blues) as of now since these are five of the popular music genres of 21st century. Also, only five music genres out of 10 have been selected for better accuracy and performance of the system.

3.1.3 Data preprocessing

Since Python has better analysis libraries for .wav files in comparison to .au files and since .wav is lossless music file format for better analysis, .au files in GTZAN music dataset are converted into .wav file format for implementation of further phases in the development of the system.

3.2 Algorithm

3.2.1 Support Vector Machine (SVM)

Support Vector Machine is state-of-the-art supervised machine learning algorithm. Though SVM is a new machine learning algorithm, it is very popular and has been used in many research works concerning not only classification problems, but also other machine learning applications such as pattern recogntion (hand writing recognition).

The basic SVM is a binary linear classifier which chooses a hyperplane representing largest separation, called margin, between the two classes based on support vectors. Support vectors are the data points that lie closest to the decision surface and are most difficult to classify.
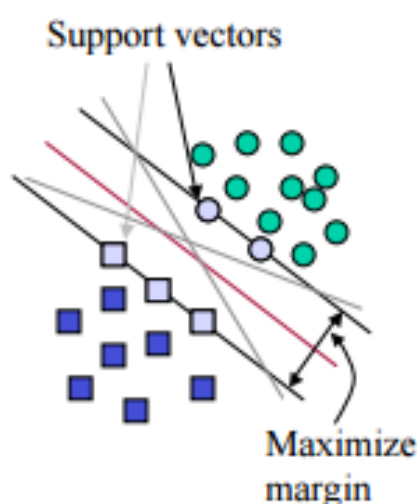
Figure 4 - General idea behind binary linear SVM

If such separating hyperplane does exists, it is known as maximum margin hyperplane and the classifier is known as maximum margin classifier. If such hyperplane does not exist, soft margin method, the one that finds hyperplane splitting the data points of two classes as distinctly as possible with maximum distance between the support vectors, is made use of.

For data points that are not linearly separable in the input space, SVM offers a possibility to find solution by making a non-linear transformation of original input space to high

dimensional space where optimum separating hyperplane can be found. This can be done through SVM Kernel.

Binary SVM

<u>Input</u>: The input of binary SVM is a set of (input, output) training pair samples; in the ordered pair, input signifies sample features x1, x2...xn, and the output signifies corresponding label, y. Typically, there can be lots of input features xi.

<u>Output</u>: set of weights Wi, one for each feature, whose linear combination predicts the value of y.

In Automatic Music Genre Classifier, the weight vector is obtained by solving the equation XW=Y, where Y includes the corresponding genre labels, one for each music file (that is, for each row of X).

$$
X = \begin{bmatrix} X_0^{(1)} & X_1^{(1)} & X_2^{(1)} & \cdots & \cdots & X_{40944}^{(1)} \\ X_0^{(2)} & & & & & X_{40944}^{(2)} \\ X_0^{(3)} & & & & & X_{40944}^{(3)} \\ X_0^{(4)} & & & & & X_{40944}^{(4)} \\ \cdot & & & & & \cdot \\ \cdot & & & & & \cdot \\ \cdot & & & & & \cdot \\ X_0^{(160)} & & & & & X_{40944}^{(160)} \end{bmatrix}_{160 \times 40944}
\qquad
W = \begin{bmatrix} W^{(1)} \\ W^{(2)} \\ W^{(3)} \\ \cdot \\ \cdot \\ \cdot \\ W^{(40942)} \\ W^{(40943)} \\ W^{(40944)} \end{bmatrix}_{40944 \times 1}
$$

The size of Feature Vector is 160 x 40944. Consider the following:

Table 2 - Tables explaining the size of Feature Vector fed into each binary SVM

Genre	Total samples for binary SVM=160
Genre A	80
Genre B	80

Feature	Number of features =40944
FFT	2048
MFCC	2922 * 13 =38896

Note: There are 2922 frames of MFCC with 13 coefficients in each frame.

Optimization of maximizing the margin is used to reduce the number of weights that are nonzero to just a few that correspond to the important features that 'matter' in deciding the hyperplane. These nonzero weights correspond to the support vectors (because they 'support' the separating hyperplane).

SVM can be used for multiclass classification as well by combination of multiple such binary SVMs. There are two approaches for it: one-versus-one (pairwise) approach and one-versus-rest approach. A potential problem with one-versus-rest is that when the number of classes is large, each binary classification becomes highly unbalanced. Considering scalability, which is one of the non-functional requirements of this project, one-versus-one approach has been used.

One-Versus-One SVM

In One-Versus-One SVM, a total of $[k*(k-1)]/2$ individual binary classifiers are used, where k indicates number of classes. For this project, $[k*(k-1)]/2 = 10$ since k=5. Applying each binary classifier to the test input gives one vote to the winning class. The test input is then labeled to the class with the most votes.

3.2.2 Windowing

When a continuous signal is converted into discrete form, certain error is resulted. The error is termed as quantization error. Any fourier transform assumes that the input signal is periodic. If the signal falls smoothly to zero at each end of the measurement interval, such signal will have no discontinuities. In our case, we have considered the first 30 seconds of music files as our dataset as in the form present in GTZAN dataset. Hence, in some cases the signal falls smoothly to zero at each end while it some cases it does not. Some cases of discontinuities is thus implied. Spectral leakage is resulted due to such discontinuities. Spectral leakage shows non-zero frequency for zero valued frequency. Windowing helps to mitigate quantization error and spectral leakage.

Hamming window function is used:

$$y[n] = x[n] * h[n]$$

$$h[n] = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N-1}\right); 0 \le n \le N-1$$

y[n] is the resulting vector of signal data.

x[n] is the input signal data.

h(n) is the hamming function and N= total number of values

3.2.3 Fast Fourier Transform (FFT)

The time-amplitude plot of a signal does not provide us with much information. Important information is encoded in the frequency, phase and amplitude of the spectral components of signal, rather than in its amplitude. Waveform is disassembled into spectral components using Fourier Transform, similar to process that happen in human auditory system.

Of the various Fourier Transform techniques, faster version of one-dimensional DFT, that is FFT, is used. The following formula is implemented for one-dimensional FFT calculation:

$$F(u) = (1/2)[(1/K) \sum f(2x) * W_{2K}^{u(2x)} + (1/K) \sum f(2x+1) * W_{2K}^{u(2x+1)}]$$

where, $\sum$ is from x=0 to x=K-1; u=0,1,…, M-1;

M=2K= total number of values

F(u) gives frequency value which is a complex number

f(2x) and f(2x+1) give amplitude values

$W_{2K} = W_M$ where $W_M = e^{-i*2*\pi}$ and $e^{-i*2*\pi} = \cos(2\pi)-i*\sin(2\pi)$

Windowing is carried out on the signal before FFT calculation.

3.2.4 Mel-Frequency Cepstral Coefficient (MFCC)

Timbre is the quality of musical note, sound or tone that distinguishes different types of sound productions, such as voices and musical instruments, string instruments. The

Acoustical Society of America (ASA) defines timbre as "the attribute of auditory sensations that enables a listener to judge two non-identical sounds, similarly presented and having the same loudness and pitch and are dissimilar". This timbral feature is represented by MFCC.
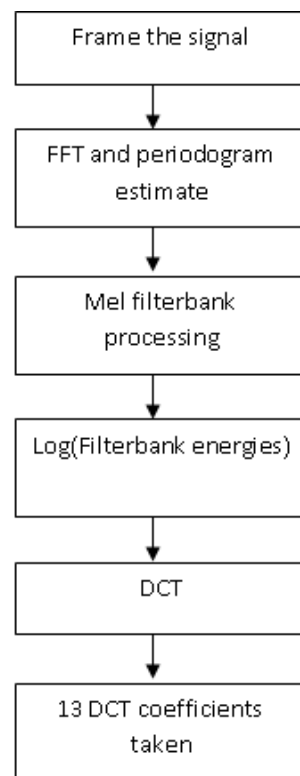
Figure 5 - Overview of the process of MFCC

The 30 seconds music clips of GTZAN music dataset has sampling frequency of 22050 Hz, meaning 22050 samples in a second.

Steps:

a) Divide the signal into frames, with following configuration:

Frame length= 25ms= 0.025 s =0.025*22050=551.25 (approx 551) samples

Frame step= 10ms= 0.010 s= 0.010*22050=220.5 (approx 221) samples

b) For each frame:

Apply windowing function to each sample of frame of signal $S_i(x)$ where i=frame number.

Perform one dimensional FFT of size 512 to get $S_i(k)$.

$P_i(k)=(1/N)* |S_i(k)|^2$ which is the periodogram estimate.

Size of $P_i(k)$ is 1x257 since we take only first 257 coefficients; the last 255 are redundant due to nature of Fourier transform (conjugate symmetry).

c) Apply Mel filterbank processing, 26 Mel filterbanks. Mel filterbank processing helps to convert power spectrum to Mel spectrum. Each of the 26 Mel filters of size 257 is multiplied with $P_i(k)$, summing 257 coefficients in each case. So, we have 26 filterbank energies. 26 Mel filterbanks is calculated as follows:

  i. Find highest and lowest frequency

  highest=samplerate/2

  lowest=0

  ii. Convert them to Mel

  $M(f)=1125 \ln(1+f/700)$

  iii. Compute values evenly spaced in Mels (in the spacing of 2)

  iv. Convert them to hertz to get h(i).

  $M^{-1}(m)=700 (\exp(m/1125)-1)$

  v. Convert them to FFT bins.

  $f(i) = floor((nfft+1)*h(i)/samplerate)$

  vi. Create filterbanks.

  $H_m(k) = 0$ for $k < f(m-1)$ and $k > f(m+1)$
  $H_m(k) = [k - f(m-1)] / [f(m) - f(m-1)]$ for $f(m-1)<= k <=f(m)$
  $H_m(k) = [f(m+1) - k] / [f(m+1) - f(m)]$ for $f(m)<= k <=f(m+1)$

, where M is the number of filters we want, f() is the list of m+2 Mel-spaced frequencies, and $1<=k<=257$ since size of $P_i(k)$ is 1x257 because we take only first 257 coefficients due to conjugate symmetry of Fourier Transform.

d) Take the log of each of the 26 energies from step c.

e) Find DCT to de-correlate overlapping energies.

  $x_k = \sum (x_n * \cos[(\pi/N)*(n+0.5)*k])$ for k=0,…,N-1
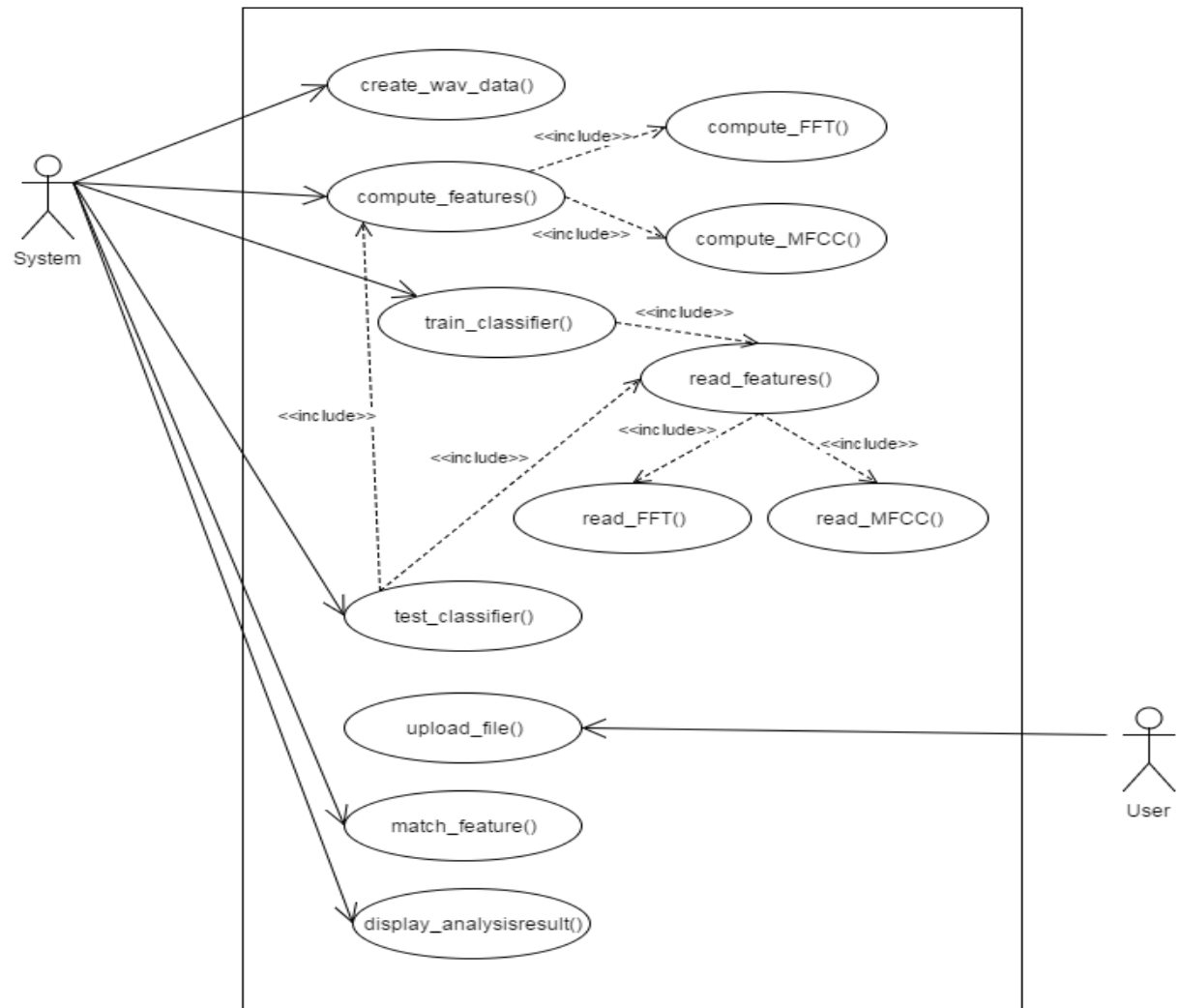 where, $\sum$ is from n=0 to n=N-1

f) Take 13 out of 26 coefficients.

3.3 System Design

3.3.1 System Use Case Diagram

Figure 6 - Use Case Diagram of Automatic Music Genre Classifier

The use case diagram in Figure 6 shows that prominent role is played by the system, which carries out all the back end tasks. The role of user is to upload music file to get to know its genre.
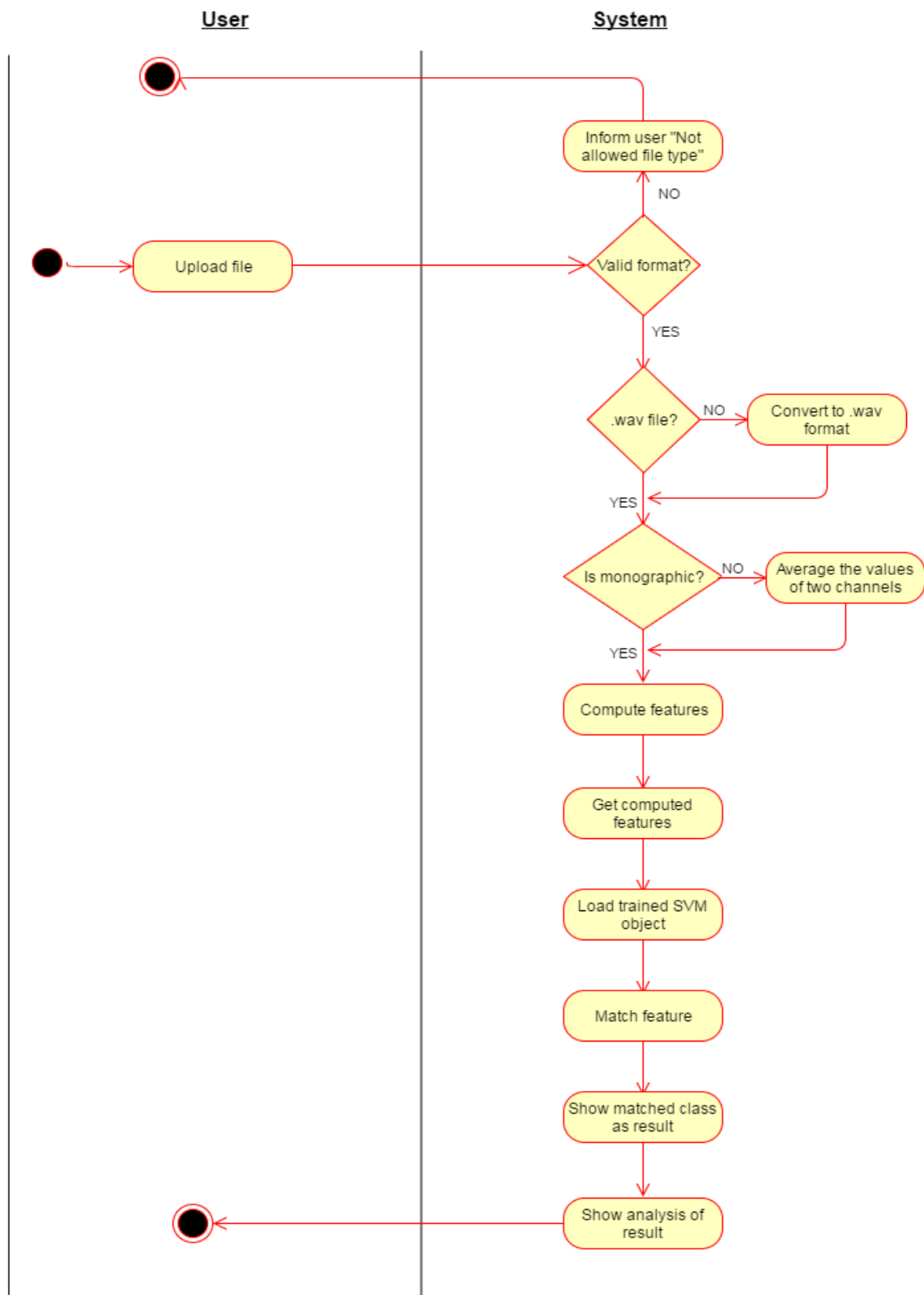
3.3.2 System Activity Diagram

Figure 7 - Main Activity Diagram of Automatic Music Genre Classifier

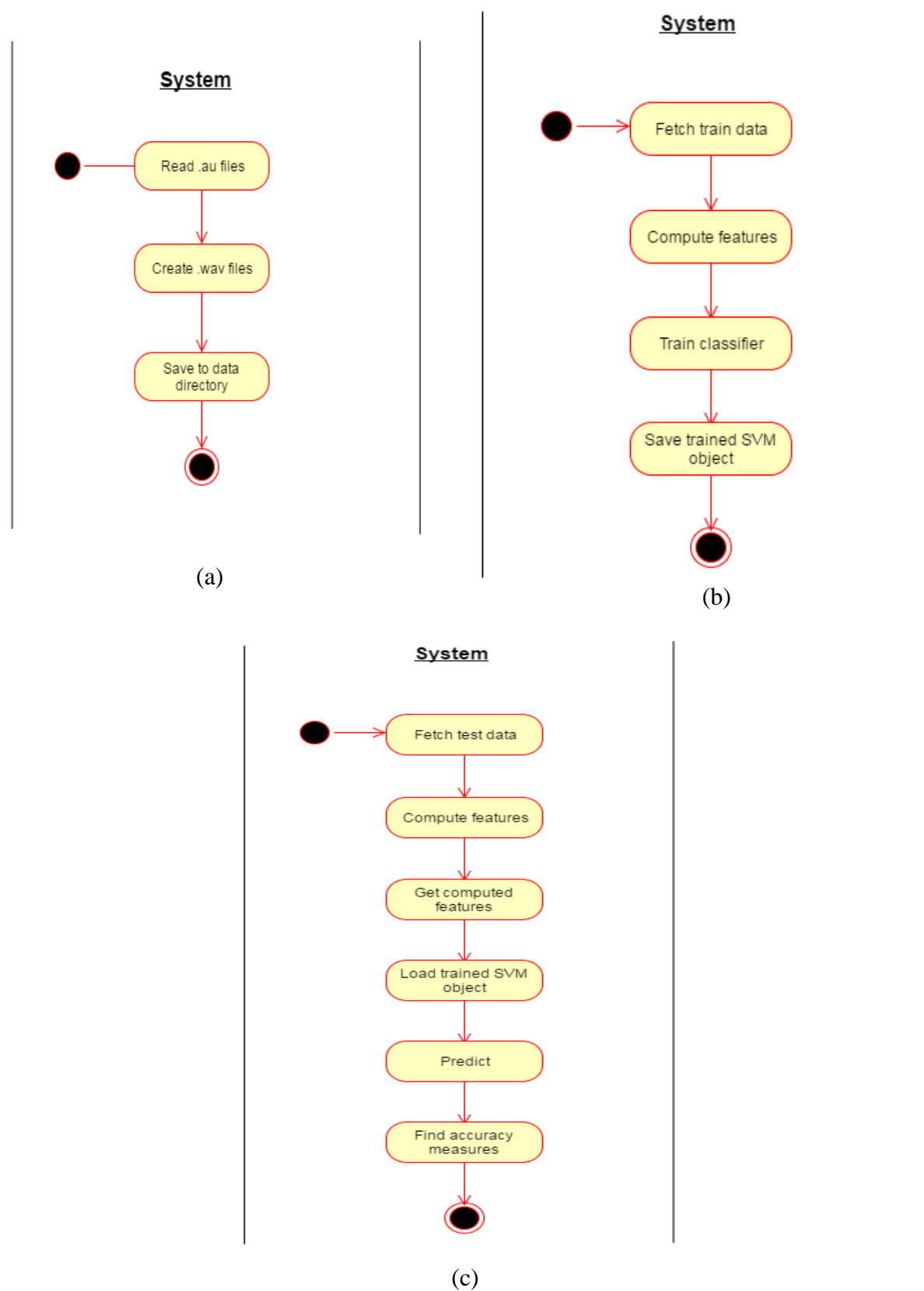**3.3.3 Supporting Activity Diagrams**

(a)

(b)

(c)

Figure 8 - Activity Diagrams (a) Create Data   (b) Training   (c) Testing

CHAPTER 4: IMPLEMENATION AND TESTING

4.1 Implementation

Automatic Music Genre Classifier has the ability to predict the class or label of the music file that has been provided to the system. For music classification, we need to implement the machine learning algorithm which is implemented by using a popular machine learning module written in Python built on SciPy.org known as Scikit-Learn. Scikit-Learn is used in this project because it is a simple and efficient tool for data mining and data analysis with different inbuilt and popular machine learning algorithms. Also, it is accessible to everybody, and is reusable in various contexts. It is commercially usable with BSD license. Similarly, feature extraction (FFT and MFCC) was based on SciPy.org and Python Speech Features, which are two popular Python modules.

### 4.1.1 System Abstract Workflow
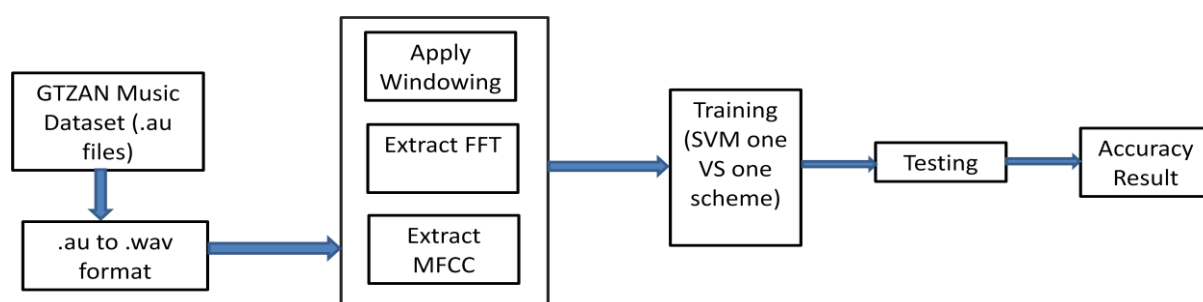
Figure 9 – Backend Workflow of Automatic Music Genre Classifier

Figure 9 explains the backend workflow of Automatic Music Genre Classifier. First, the GTZAN Music Dataset is converted to wav format. The wav files are read; sampling is performed at 22050 Hz. Windowing is performed on the sample points of each wav file. FFT calculation is performed to find out 2048 FFT values and MFCC calculation is done to find out 2292 frames each with 13 coefficients, hence making feature vector of size 160 x 40944 (160 music files and 40944 feature values).

During the initial stage of project implementation, in order to avoid overburdening the classifier, only 2061 feature values were used: 2048 FFT values and 13 MFCC values averaged over the frames. However, due to averaging, the accuracy obtained was only 62.0% and the confusion matrix was not satisfactory. Figure 10 shows the confusion matrix and accuracy so obtained.
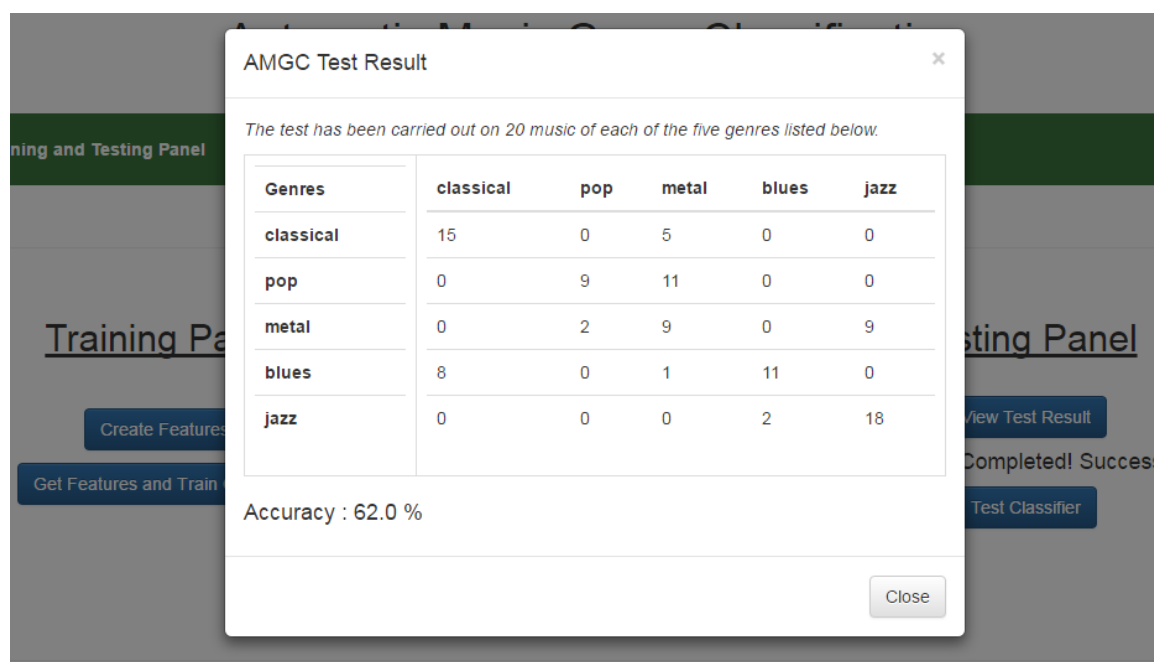
Figure 10 – Result obtained when averaging the MFCC values over frames

Since the accuracy and confusion matrix were not satisfactory, the 13 coefficients of all the 2922 frames were used, along with 2048 FFT values, to create feature values of size 40944.

Continuing with the backend workflow, feature vector is then fed to training algorithm to find trained model which is then tested and ready for use in the user module.

The backend task is carried out by the Admin accessing "/adminDashboard" in the root URL, and entering the admin key. The admin can train and test the system.

The user can access the root URL (without need for any key or password) to upload a music file and view its analysis and genre tag. The workflow is shown in the activity diagram in Figure 7.

4.1.2 Tools Used

CASE tools:

a) Draw.io

Client Side:

1. HTML is used to display content in the browser.
2. CSS is used to adjust the layout, look and design of the HTML content.
3. Bootstrap CSS framework is used for beautifying the HTML elements to improve the user experience.

Server Side:

1. Python programming language is used to implement the core program logic.
2. Flask web framework is used for dynamic webpage generation and to display the predicted result in the browser as well as to handle page requests for music file upload.
3. SciKit-Learn is used to implement the machine learning algorithm.
4. SciPy and Python Speech Features are used to read and process music files, and extract features.

4.2 Description of Major Classes

The major classes in the application are:

4.2.1 SVM

This class is responsible for the implementation of one-versus-one SVM for classification. We consider two important attributes/parameters: Kernel and probability.

Linear kernel has been chosen since we have large number of features and since linear SVM is less prone to overfitting than non-linear, and probability has been set to true (in order to view prediction probabilities). Three important SVM methods are as follows:

1. fit(X, y, sample_weight=None)

   Here, X is the training input samples. It is a matrix of input datasets which is feature of the image. y is the target values (class labels). It is a vector (1

Dimensional). Sample_weight is optional. If sample_weight is not provided then the sample weights are initialized to 1/n_samples. This method builds a supervised one-versus-one SVM classifier from the training set (X, y).

2. predict(X)

This method predicts classes for X (the training input samples or matrix). This method returns the predicted classes y (the output vector).

3. predict_proba(X)

This method predicts class probabilities for X. This method returns the list of class probabilities of the input samples. The output is a two dimensional array of size num_input_files by num_of_classes. This method can be used only when probability attribute is set to true.

4.2.2   MusicFeatureExtraction

This class is responsible for extracting features from the music files present in training and testing sets. The important methods of this class are:

1. computeFeatures(key)

'key' indicates whether we are carrying out training or testing for fetching correct collection of music files placed in training or testing directory accordingly.

ComputeFeatures is responsible for computing FFT and MFCC values accordingly and saving them to respective training and testing directories for FFT and MFCC values. For example, for training case:

```
FFT_dir="D:\\MiniProject\\genres_fft_csv\\<key>"
MFCC_dir="D:\\MiniProject\\genres_mfcc_csv\\<key>"
```

<key> can be train and test for training and testing respectively.

For example, for training case:

```
FFT_dir="D:\\MiniProject\\genres_fft_csv\\train"
MFCC_dir="D:\\MiniProject\\genres_mfcc_csv\\train"
```

FFT and MFCC values so computed are saved so that they do not need to be computed every time for training the system. They need to be computed only when new music files are added to training or testing directories.

2. readFeatures(key)

'key' indicates whether we are carrying out training or testing for fetching correct collection of music files' features placed in FFT_dir and MFCC_dir directory accordingly.

It reads the features saved during computeFeatures(key). This method hence makes use of computeFeatures(key) method.

3. getFeatures(key)

This method makes use of readFeatures(key) method. This method forms feature vector by combining the values of FFT and MFCC which are read from readFeatures(key). The output of this is fed as input samples X in the training algorithm.
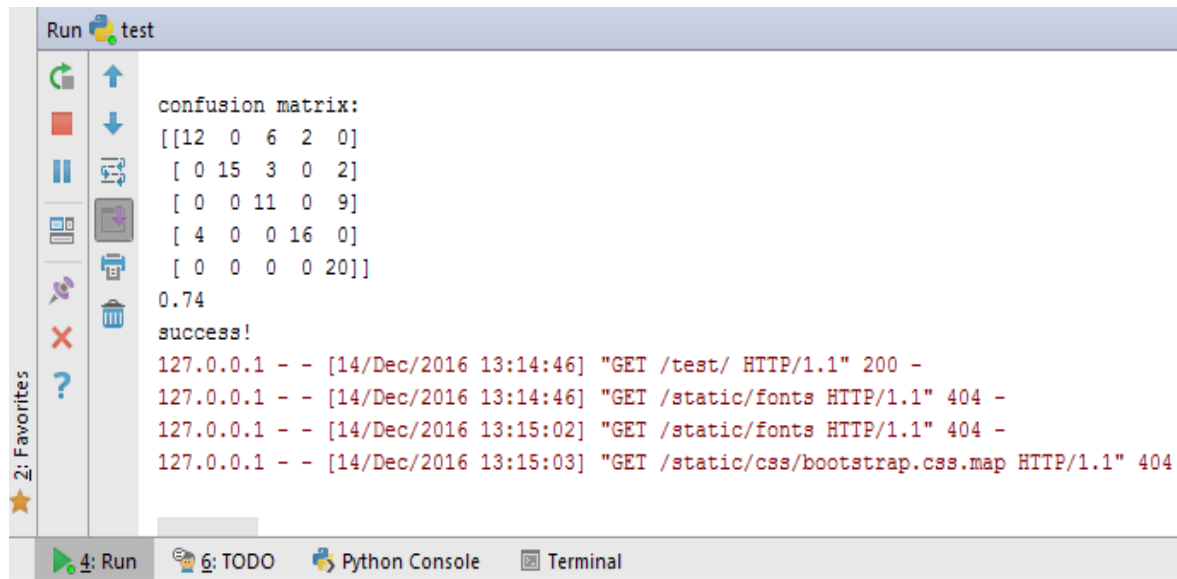
4.3 Testing

4.3.1 Overall System Testing

During overall system testing, 20 music files of each of the five genres were made use of. MusicFeatureExtraction class and its methods were used to create feature vector that was then used to predict the respective genres of the test music files.

Confusion matrix was calculated in order to find out actual and predicted classifications done by a classification system. Following confusion matrix was obtained.
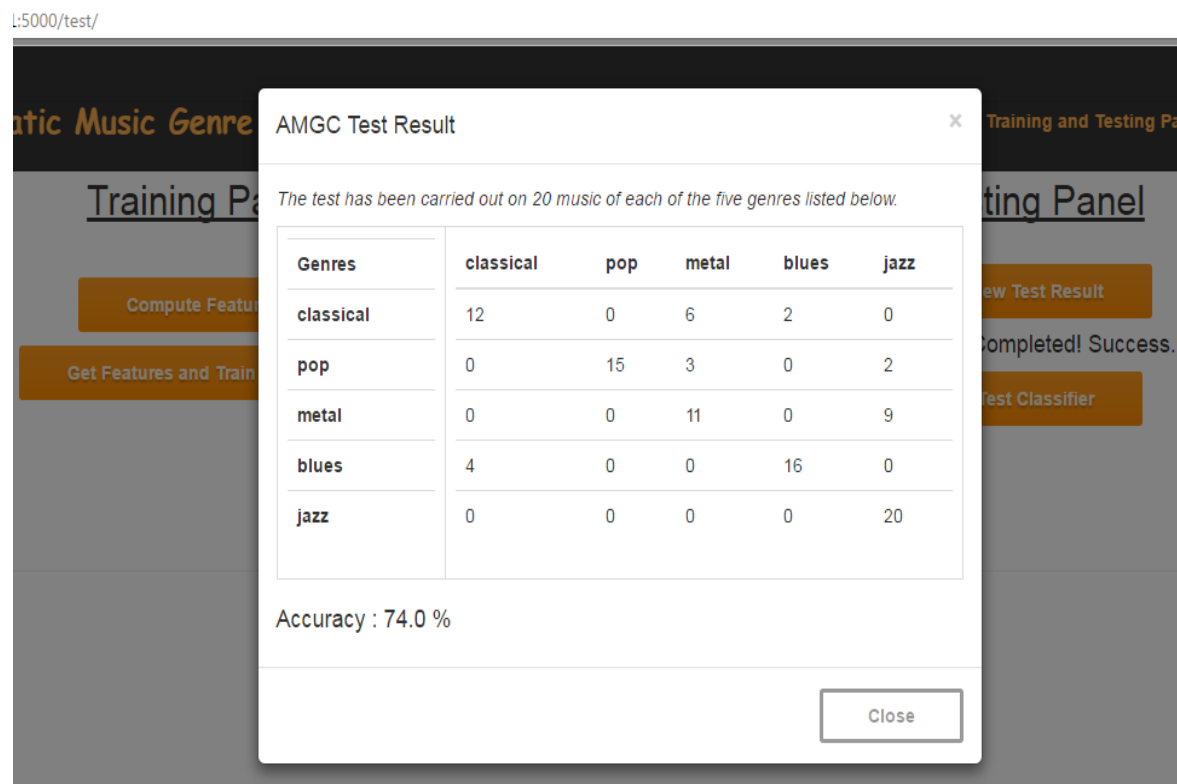
(a)

(b)

Figure 11 - Confusion Matrix – (a) Backend View | (b) Front End View

Figure 11 indicates the Confusion Matrix. Out of 20 music files of each genre, the diagonal in the confusion matrix shows the count of correctly classified music files.

31

4.3.2 Testing for Non-Allowed File Format

TC01- Music file of format besides mp3, wav, aiff, m4a, and au upload
Precondition: Music file is available in the local storage
Assumption: Music file is not corrupted
Test steps 1. Navigate to /userPanel page 2. Choose music file of format besides mp3, wav, aiff, m4a, and au 3. Click on Process button
Expected Result: Error message "File Type Not Allowed" should be displayed
Generated Result: Error message "File Type Not Allowed" displayed. Result is in Appendix A (Figure 14).

4.3.3 Displaying Prediction Result on Web Interface

TC02- Result display on the web interface
Precondition: Uploading music file is carried out successfully by the system
Assumption: The system is working without any error and music file has been uploaded
Test steps 1. Navigate to /userPanel page 2. Choose music file of format mp3, wav, aiff, m4a, and au 3. Click on Process button
Expected Result: Rendered result should contain predicted genre label, accuracy and four analysis graphs (signal, FFT, MFCC, spectrogram)
Generated Result: Predicted label displayed successfully with accuracy and four analysis graphs (signal, FFT, MFCC, spectrogram). Result is in Appendix A (Figure 15).

CHAPTER 5: MAINTENANCE AND SUPPORT PLAN

5.1 Maintenance Plan

Automatic Music Genre Classifier will implement corrective maintenance for resolving different bugs and errors that may occur when this project is made live. Perfective maintenance will be implemented for increasing efficiency of the system by optimizing various implementation methods. Preventive maintenance will be implemented to make sure that the system will not be harmed by hackers and security mechanism will be added.

5.2 Support Plan

Automatic Music Genre Classifier will be presented to the respective authority of different music schools in the country and to research units in order to help the project grow.

CHAPTER 6: CONCLUSION AND RECOMMENDATION

6.1 Conclusion

The system developed, Automatic Music Genre Classifier, that makes use of FFT and MFCC for feature extraction and SVM for classification has been successfully implemented using Python programming language and Flask Framework. An accuracy calculated as correct count/total count was found to be 74.0%

6.2 Recommendation

The accuracy can be increased above 74.0% by consideration of additional features such as beat (calculated with the use of FFT), pitch and a better algorithm over FFT for non-stationary music files, which is Discrete Wavelet Transform (DWT). Moreover, instead of using 30 seconds music files of GTZAN Music Dataset, more time duration of the music files can be used.
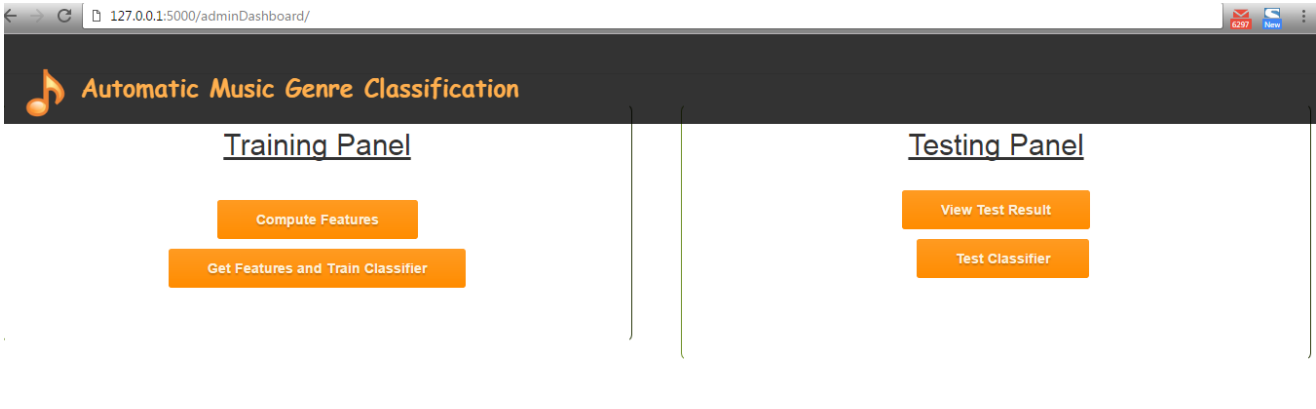
APPENDIX A
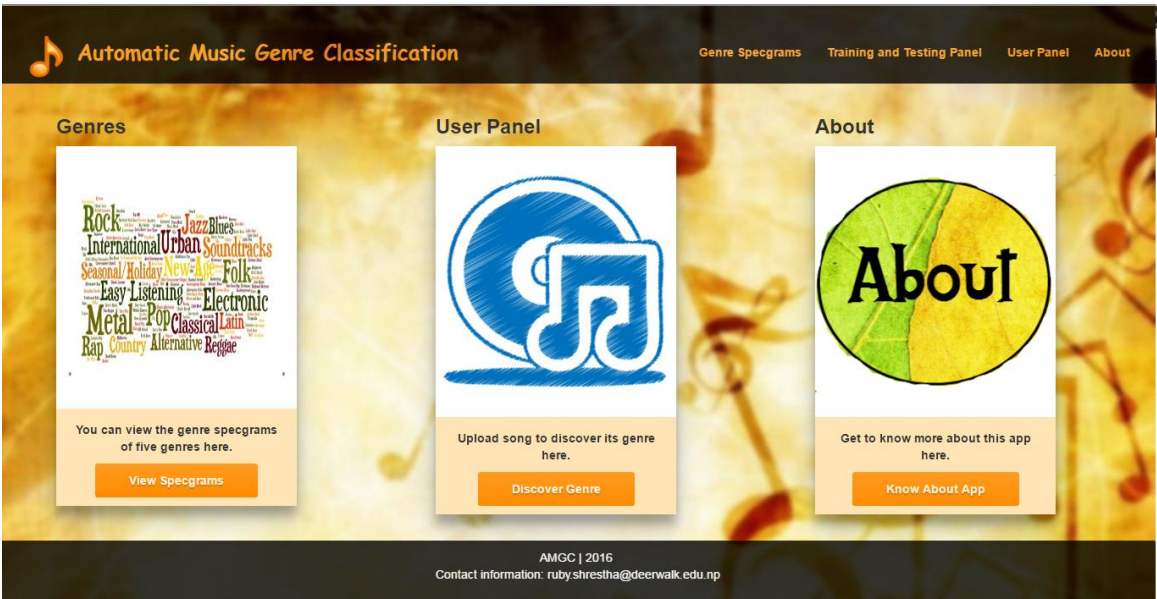

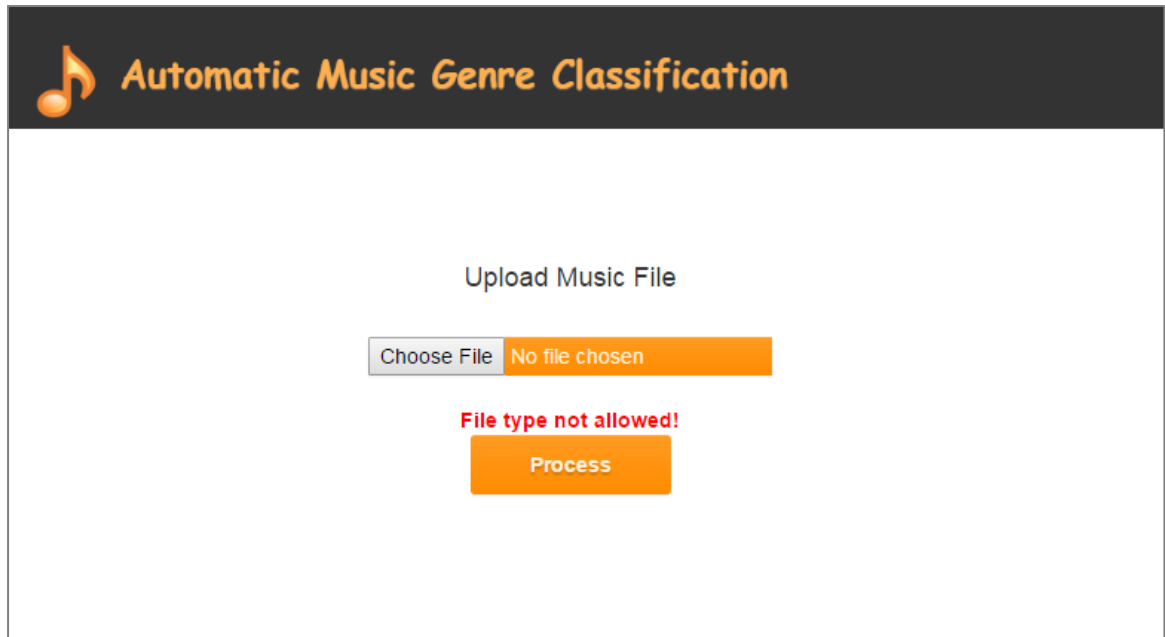**Figure 12 - Admin View of the Project**

Figure 13 - User View of the Project

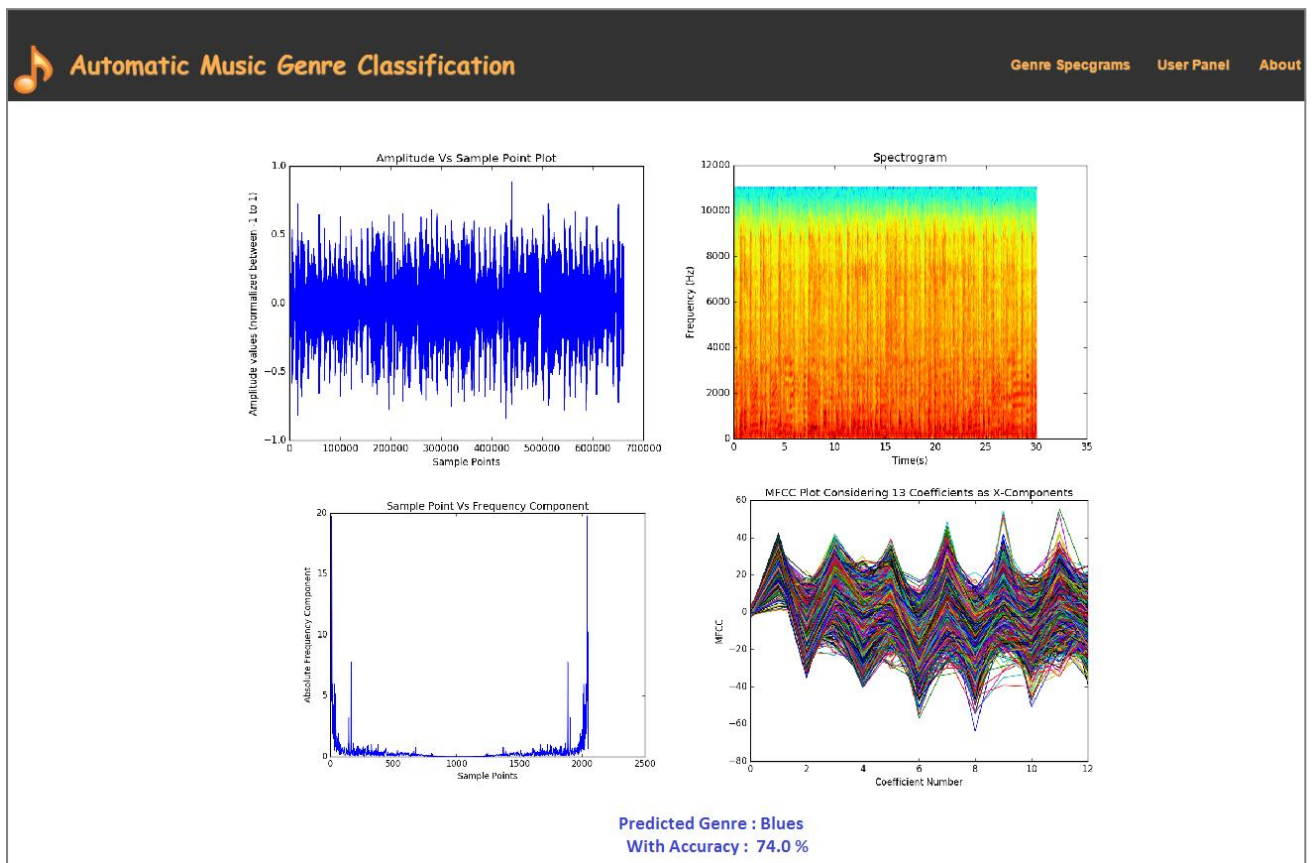**Figure 14 - Test Case TC01 Screenshot**

Figure 15 - Analysis Result with Predicted Genre Label and Accuracy

APPENDIX B

Example Calculations

Suppose the music amplitude values are:

x=[1.5, 3.5, 2.75, 6.5, 1.0, 5.25]

Implementing Hamming Window Function

Algorithm:

```
function get_window(N):
    window=empty list
    loop i from 0 to N:
        intermediate=(2*pi*i)/(N-1)
        val=0.54-0.46*cos(intermediate)
        window.add(val)
    return window
```

For the example case, N=len(x)=6

hamming_window_values = [0.08000000000000002, 0.3978521825875242, 0.9121478174124757, 0.9121478174124759, 0.3978521825875243, 0.08000000000000002]

Performing Windowing:

mData=x* hamming_window_values = [0.120, 1.392, 2.508, 5.928, 0.397, 0.420]

FFT Calculation

FFT calculation is done on the values obtained after windowing is performed. In this case, FFT calculation is done on mData.

Given,

M=6 => 2*K=6 => K=3

u=0, 1, 2, 3, 4, 5

Calculating,

$F(0) = (1/2)[(1/3) \sum (f(2x) * e^{-i*2*\pi*u(2x)}) + (1/3) \sum (f(2x+1) * e^{-i*2*\pi*u(2x+1)})]$

where $\sum$ is from 0 to K-1, that is, 0 to (3-1)=2

$= (1/2)[(1/3) * (f(0) * e^{-i*2*\pi*0 * (2*0)} + f(2) * e^{-i*2*\pi*0 * (2*2)} + f(4) * e^{-i*2*\pi*0 * (2*4)}) + (1/3) * (f(0) * e^{-i*2*\pi*0 * (2*0)} + f(2) * e^{-i*2*\pi*0 * (2*2)} + f(4) * e^{-i*2*\pi*0 * (2*4)})$

$= 10.76770213 + 0.j$

Similar calculations are performed for F(1) to F(5) to get the following FFT values as result:

FFT(x) = [10.76770213+0.j ; -6.35584883-2.66998832j ; 3.68959015+0.98559898j ;

-4.71518477+0.j ; 3.68959015-0.98559898j ; -6.35584883+2.66998832j]

Calculating Mel filterbanks

Mel filterbank calculation is a part of MFCC calculation.

Suppose,

Sample rate = 16000 Hz

Lowest Frequency= 300 Hz

Highest Frequency= sample rate / 2 = 8000 Hz

Using equation:

$M(f) = 1125 \ln(1+f/700)$

, the upper and lower frequencies are converted to Mels. Here, 300Hz is 401.25 Mels and 8000Hz is 2834.99 Mels.

For instance, 10 filterbanks are considered, for which 12 points are required. This means 10 additional points spaced linearly between 401.25 and 2834.99 are required. Hence,

m(i) = 401.25, 622.50, 843.75, 1065.00, 1286.25, 1507.50, 1728.74,

1949.99, 2171.24, 2392.49, 2613.74, 2834.99

Converting back to hertz, using formula $M^{-1}(m) = 700 (\exp(m/1125)-1)$,

h(i) = 300, 517.33, 781.90, 1103.97, 1496.04, 1973.32, 2554.33,

    3261.62, 4122.63, 5170.76, 6446.70, 8000

First and last frequencies are the values initially supposed as lowest and highest frequencies respectively.

Now, the frequencies require to be converted to nearest frequency bins. This process does not affect the accuracy of the features. To convert the frequencies to FFT bin numbers, the FFT size and the sample rate needs to be known. Let, FFT size=512 and sample rate, as already supposed, is 16000 Hz.

f(i) = floor((nfft+1)*h(i)/sample rate)

This results in the following sequence:

f(i) = 9, 16, 25, 35, 47, 63, 81, 104, 132, 165, 206, 256

Now, these frequency bins can be used to create filterbanks using the following formula.

$H_m(k) = 0$ for $k < f(m-1)$ and $k > f(m+1)$

$H_m(k) = [k - f(m-1)] / [f(m) - f(m-1)]$ for $f(m-1) <= k <= f(m)$

$H_m(k) = [f(m+1) - k] / [f(m+1) - f(m)]$ for $f(m) <= k <= f(m+1)$

Here, $1 <= m <= M$, M=total number of filterbanks required.

In case of this project, $1 <= k <= 257$ since size of $P_i(k)$, power spectrum values obtained in MFCC calculation, is 1x257, because we take only first 257 coefficients due to conjugate symmetry of Fourier Transform.

REFERENCES

Adami, A. G. (2010). Automatic Speech Recognition: From the Beginning to the Portuguese Language. *The Int. Conf. on Computational Processing of Portuguese (PROPOR).* Rio Grande do Sul: Porto Alegre.

Ali, M., Hassain, M., & Bhuiyan, M. N. (2013). Automatic Speech Recognition Technique for Bangla Words. *International Journal of Advanced Science and Technology, 50.*

Annesi, P., Basili, R., Gitto, R., & Moschitti, A. (n.d.). *Audio Feature Engineering forAutomatic Music Genre Classification.* Colombo.

Anusuya, M. A., & Katti, S. K. (2009). Speech Recognition by Machine: A Review. *International Journal of Computer Science and Information Security, 6*, 181-205.

C., V., & V., R. (2015). Isolated Speech Recognition System For Tamil Language Using Statistical Pattern Matching And Machine Learning Techniques. *Journal of Engineering Science and Technology*, 617-632.

Casey, M. A. (2008). *Content-Based Music Information Retrieval:Current Directions and Future Challenges.*

Coelho, L. P., & Richert, W. (n.d.). *Building Machine Learning Systems with Python* .

Correa, D. C., Costa, L. d., & Saito, J. H. (2010). Tracking the Beat: Classification of Music Genres and Synthesis of Rhythms. *17th International Conference on Systems, Signals and Image Processing*, (pp. 280-283). Sao Carlos.

Dharun, V. S., & Karnan, M. (2012). Voice and Speech Recognition for Tamil Words and Numerals. *International Journal of Modern Engineering Research*, 3406-3414.

Elimat, A. K., & AbuSeileek, F. A. (2014). Automatic Speech Recognition technology as an effective means for teaching pronunciation. *The JALT CALL Journal*, 21-47.

Fabbri, F. (1980). A THEORY OF MUSICAL GENRES:. *First International Conference on Popular Music Studies* , (pp. 1-1). Amsterdam.

Gaikwad, S. K., Gawali, B. W., & Yannawar, P. (2010). A Review on Speech Recognition Technique. *International Journal of Computer Applications, 10*, 16-24.

Goulart, A. J., Guido, R. C., & Maciel, C. D. (2012). Exploring different approaches for music genre classification. *Egyptian Informatics Journal*.

Kaminskas, M., & Ricci, F. (2012). *Contextual music information retrieval and recommendation: State of the art and challenges.*

Kosina, K. (2002). *Music Genre Recognition.* Hagenberg, Germany: Master's Thesis, Hagenberg Technical University.

Lee, J. H., & Downie, J. S. (2004). Survey of music information needs, uses, and seeking behaviours: Preliminary findings. *ISMIR*.

Li, T., Ogihara, M., & Li, Q. (2003). A Comparative Study on Content-Based Music Genre. *SIGIR Forum (ACM Special Interest Group on Information Retrieval)*, 282-289.

Logan, B. (2000). Mel Frequency Cepstral Coefficients for Music Modeling. *ISMIR*.

McKay, C., & Fujinaga, I. (2006). Musical genre classification: Is it worth pursuing and how can it be improved? 2-2.

Mishra, N., Shrawankar, U., & Thakare, D. M. (2013). Automatic Speech Recognition Using Template Model for Man-Machine Interface. *Proceedings of the International Conference ICAET 2010.* Chennai, India: ICAET.

Ms Savitha, & Upadhya, S. (2013). Digit Recognizer Using Single and Average Template Matching Techniques. *International Journal of Emerging Technologies in Computational*, 357-362.

(n.d.). Music Information Retrieval. In *CC346: Topics in Sound and Music* (pp. 1-2). University of London's External System degree programme in Creative Computing.

Neri, A., Cucchiarini, C., & Strik, W. (2003). Automatic Speech Recognition for second language learning: . *Proceedings of 15th International Conference of Phonetic Sciences*, (pp. 1157-2260). Barcelona.

PPL, PRS. (n.d.). *news-and-charts/news*. Retrieved from www.musicworksforyou.com: http://www.musicworksforyou.com/news-and-charts/news/177-how-our-brains-process-music

Stanford. (2012, July 13). *Music*. Retrieved from plato.stanford.edu: https://plato.stanford.edu/entries/music/#1

Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 293 - 302.

Tzanetakis, G., Essl, G., & Cook, P. (2001). Automatic Musical Genre Classification Of Audio Signals. *IEEE Transactions on Speech and Audio Processing*.

Wiering, F. (2006). *Can Humans Benefit From Music Information Retrieval.* Utrecht.